An Empirical Evaluation of the Performance of Real-Time Illumination Approaches: Realistic Scenes in Augmented Reality

A'aeshah Alhakamy[1,2(✉)] and Mihran Tuceryan[1(✉)]

[1] Indiana University - Purdue University (IUPUI), Indianapolis, IN 46202, USA
aalhakam@iupui.edu, {aalhakam,tuceryan}@iu.edu
[2] University of Tabuk, Tabuk, TA, Saudi Arabia

Abstract. Augmented, Virtual, and Mixed Reality (AR/VR/MR) systems have been developed in general, with many of these applications having accomplished significant results, rendering a virtual object in the appropriate illumination model of the real environment is still under investigation. The entertainment industry has presented an astounding outcome in several media form, albeit the rendering process has mostly been done offline. The physical scene contains the illumination information which can be sampled and then used to render the virtual objects in real-time for realistic scene. In this paper, we evaluate the accuracy of our previous and current developed systems that provide real-time dynamic illumination for coherent interactive augmented reality based on the virtual object's appearance in association with the real world and related criteria. The system achieves that through three simultaneous aspects. (1) The first is to estimate the incident light angle in the real environment using a live-feed 360° camera instrumented on an AR device. (2) The second is to simulate the reflected light using two routes: (a) global cube map construction and (b) local sampling. (3) The third is to define the shading properties for the virtual object to depict the correct lighting assets and suitable shadowing imitation. Finally, the performance efficiency is examined in both routes of the system to reduce the general cost. Also, The results are evaluated through shadow observation and user study.

Keywords: Direct illumination · Indirect illumination · Augmented reality · Image-based lighting · Incident light · Reflected light

1 Introduction

The illumination information extracted from the physical world can provide the means to realistically render augmented objects into the final scene. Acquiring an illumination model featuring accurate precision that would capture the

L. T. De Paolis and P. Bourdot (Eds.): AVR 2019, LNCS 11614, pp. 179–195, 2019.
https://doi.org/10.1007/978-3-030-25999-0_16

whole real environment can be challenging under reduced assumptions. A photo-realistic and dynamic augmented reality scene that integrates the illumination model requires addressing several aspects in each frame. In this paper, we evaluate two of our previous developed methods in details [23,24] where both extracted the light information from the real environment producing an illumination model that is applied onto the virtual objects to render a visually cohesive final scene. Both routes include three main aspects of the system but have different reflected light simulation method, however, they both have similar obligation to operate simultaneously in real-time. The system assumes a live-feed 360° camera is instrumented on any AR device (e.g., head mounted display, projection display, handheld mobile, or webcam camera).

The first (1) aspect of the system is to estimate the angle of incident light (i.e. direct illumination). The incident light is known as the light falling from the source onto the objects directly and is then depicted by the eyes of the observer. This estimation utilizes the 360° camera to capture the entire environment map of the real scene.

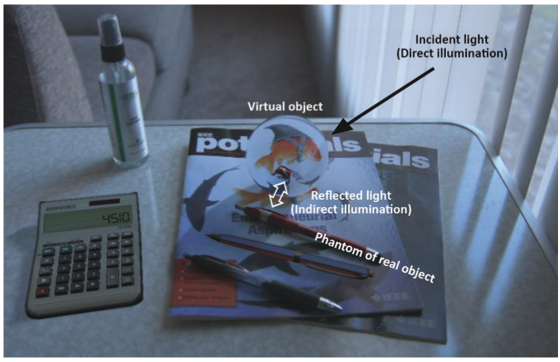

Fig. 1. Illustration of the incident light and the reflected light differences and their interactions with the real and virtual objects.

The second (2) aspect simulates the reflected light (i.e. indirect illumination) captured from the surroundings of the real world onto the virtual object. The reflected light is defined as the inter-light bouncing off the surfaces of one object into another (see Fig. 1). The indirect illumination has two separate sub-methods that are used and tested to reduce the performance cost. One of them is (a) global cube map textures captured from 360° camera view, while the other one (b) samples the local regions surrounding the virtual object form the main AR view. In the shading process these textures are added and updated as part of the image-based lighting (IBL) mode.

The third (3) aspect passes the estimated incident light and the simulated reflected light on the virtual objects through shading language and CG techniques. The environment light conditions and shadow effects from the preceding

methods can be defined through different types of shaders as required. The study aims to evaluate and explore the efficient techniques to render a realistic and coherent AR scene. The comparison process of both routes in the system would provide us with pros and cons for each method.

2 Related Work

The realistic perception of a virtual object in augmented, mixed or virtual realities has been the interest of many previous research exploration in several techniques. While the major focus of this study involves augmented reality where the virtual objects were added to the physical scene, some of the techniques used can be applicable to other fields. Although prior work assumed certain restrictions for the system to succeed, they also addressed some limitations and challenges for future work. However, their work was an inspiration for the current system.

Estimation and Detection of the Incident Light. Debevec et al. [1] presented the first state-of-the-art methods that attempt to solve the problem of the interactive rendering of a virtual object in augmented reality scenarios. Karsch et al. [2] developed a method that capture and represent part, or subset of the incident light field (ILF) for a virtual object placed in the specific region of the scene during rendering process which then facilitated the development of post-capture refocusing, depth estimation and small viewpoint transformations applications.

One of the main techniques used is spatial sampling of the incident light at multiple points which can be obtained in preprocess using the Iterated Closest Point (ICP) algorithm [17]. Besides the precomputed radiance, some assume a pre-known geometry of the real scene with more information about the angle of incident light to acquire an effective Bidirectional Reflection Distribution Function (BRDF) [7].

Furthermore, Nowrouzezahrai et al. [12] assumed the possibility of aggregating the spatial variation of the scene lighting into directional distribution represented as an environment map of the incident light. Their factorization of the light seeks to compute the dominant light direction and color separately using spherical harmonic (SH) coefficients which is obtained by a projection onto the SH basis functions. As these functions are orthonormal, the dot product was the amount of incident light from all directions at a certain vertex [15]. Gruber et al. [6] also expressed the incident light field with SH to estimate the distance of the light field which represent every incident light ray on a hemisphere from many different observers of an arbitrary geometry.

The traditional image-based lighting approach usually utilized a light probe to capture one angle in space represented in the HDR image, Unger et al. [19] used a sequence of light probes to capture a path of incident light then rendered the objects with a corresponding light probe as if illuminated by that light. Similar techniques used either a perfect reflector mirror sphere or a fish-eye camera where both could exhibit specular reflections for higher resolution. The

prefect reflector would improve additional extensions of the BRDF such as the Fresnel factor and the Schlick approximation [13,18]. In general, the development of capturing HDR images become more critical for realistic results by mining approximate illumination information directly from s video sequences to avoid physical measurements in the real scene [10].

Richter-Trummer et al. [14] used inverse rendering which factored the texture color into incident light and the diffuse albedo color through the radiance transfer method. For more material estimation Rohmer et al. [15] assumed a Lambertian environment for incident light integration over the surface of the upper hemisphere.

Stimulation of the Reflected Light. The reconstruction of the direct and indirect illumination using the radiosity algorithm for diffuse lighting on the virtual object in the real scene [4] is one of the methods to simulate the reflected light. Loscos et al. [7] also used the hierarchical radiosity to compute the indirect illumination employing a rough subdivision of the scene. The reflected light can be estimated from the photon map when the ray-tracing hit the diffuse surfaces. The rendered result composited from the real video image is directly exhibited on the output device.

The calculation of multiple bounces of indirect illumination were possible through Monte Carlo integration to evaluate the irradiances in cache records by shooting recursive rays into arbitrary directions over the surface point [8]. Metha et al. [11] used two Monte Carlo for sampling as path-tracing passes to estimate the per pixel illumination. They sampled the cosine hemisphere for indirect illumination on the diffuse surfaces on the real and virtual objects and sampled the Phong lobe for glossy surfaces on the virtual objects only. Other sampling techniques were applied on the shadow mapping to create Virtual Point Lights (VPLs) which then enables the calculation of the indirect illumination [9,15].

Global Cub Map. The cube maps have been supported as the environment map by most graphics cards to demonstrate proper global reflections with high-resolution on the virtual objects [7]. While each pixel on the unit sphere represented an individual direction, cube map were used to evaluate the SH coefficient [6]. Also, Rohmer et al. [15] rendered a low-resolution cube map from the position of the virtual object of the indirect illumination atlas as texture of the surrounding environment. Schwandt et al. [16] created a global environment map for all the virtual objects in parallel. Their calculations included both diffuse and glossy cube map for unique material properties.

Local Sampling. A 2D texture mapping was enabled by Agusanto et al. [3] which was stored in a frame buffer of an image to render the scene. A few HDR images or environment maps were used to capture the light information represented as 2D texture or 4D surface light fields then projected onto a geometric model [10]. Franke [5] formulated two shading systems to estimate the natural illumination

and compute the reflected light on several materials of the virtual objects in real-time.

Utilization of 360° Panoramic Videos. The utilization of 360° video sequence to capture the environment map is not a novel notion. Rhee et al. [20] used a conventional low dynamic range (LDR) 360° videos to render an interactive mixed reality scene. Iorns et al. [21] developed a system to use a live streaming 360° video as an input for image-based lighting where real-time shadowing and reflection were investigated. Michiels et al. [22] used an appropriate 360° environment map linked to a car position for real-time lighting added to the rendering equation.

3 Method

This section describes the entire system briefly where rendering a visually coherent final scene is the ultimate goal. The system consists of our previous and current three main methods where the second method is branched into two submethods. For a complete overview of the entire system (see Fig. 2).

1. **Estimate Incident Light.** The live-feed of the physical environment is captured using a 360° camera with the support of parallel computations to sample the light area and estimate the angle of the real light source. Then, A virtual light in the virtual scene is created to imitate the incident light.
2. **Simulate Reflected Light.** The inter-bouncing light between the objects and the surrounding is captured using two methods. The resulted texture is rendered into the virtual object based on the material properties.
   - *Global Cube Map.* The panoramic HDR of the 360° camera is used to create the 6 faces as a 2D texture to construct a cube map for the image-based lighting mode which can be modified while defining the shading properties for each virtual object.
   - *Local Sampling.* The main camera of the AR view device is employed to sample the region below and surrounding each virtual object. The resulted texture is then used in IBL mode where the material property of each object can be reflected on the objects.
3. **Define Shading Properties.** the virtual object materials and characteristics are defined in this method as needed. The object's main texture, normal map, specular map, and diffuse elements are also addressed in this method in addition to many other properties that we are going to discuss in details later.

Rendering. The global illumination for the entire system is used to create more realistic appearance. The suited rendering path for real-time lights is the deferred shading which has the best lighting and shadow reliability. The forward rendering and legacy deferred are also used when trade-offs are necessary. These rendering paths support the per-pixel lighting including normal maps and light cookies. Additional rendering passes would not be required for reflection depth and normal buffers. They support semitransparent objects and anti-aliasing. The number of pixels illuminated influence the performance cost of a per-pixel light instead of the number of lights.

Tracking. Various interactive AR/VR experiences are developed using rotational and positional trackers. The device location is relative to the physical world and provides the device tracker with positional or rotational information. We employed the Vuforia engine in our system which supports the positional device tracker. This type of tracker is suitable when content is added in the environment for a robust 6 degrees-of-freedom target tracking. However, various scripts were developed to provide additional configurations for the different light conditions while the objects, camera or marker is moving.

Hardware Description. The system operates on a personal computer with Intel®, Core™ i7-3930k CPU @ 3.20 GHz 3201 MHz, six core(s), 64.0 GB RAM, and NVIDIA GeForce GTX 970 GPU. As input devices, two cameras are used: DSLR Nikon D7200 and live-feed RICOH THETA S 360°.

Additional Software. The camera live-feed requires a broadcasting software and system registry to display the input video through the 3D engine. It is recommended to check devices compatibility while transferring the code onto different machines.

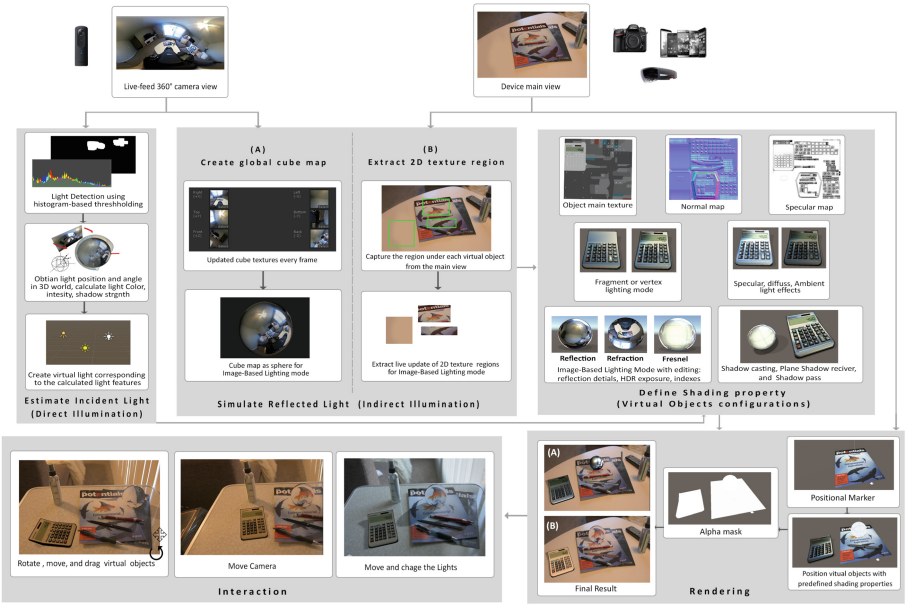

Fig. 2. A full overview of the entire system which consists of: estimate incident light, simulate reflected light, and define shading property followed by rendering and interaction.

3.1 Estimate Incident Light

The panoramic 360° view provides a complete environment map of the physical world where the illumination information can be estimated for an arbitrary number of lights. The 3D engine reads the camera feed as a 2D texture which is then converted to an OpenCV Mat object using parallel computation support, and finally converted to an OpenGL 3D vector as the 3D engine interacts with the data. This procedure is then reversed to represent the outcomes.

The estimation process samples the incident light based on the saturated bright areas of the image frame from the live-feed. The threshold is based on the histogram median of the entire frame. A suitable noise reduction through Gaussian blur is used for pure contours. Any additional noise blobs are removed by performing a series of erosion and dilation functions for specific structuring elements.

Each light is represented as a contour which is sorted based on the size to determine the main light followed by the second significant light and so on. The contour moments are used to specify the light location on the panoramic view.

The spherical projection is used to transform the screen coordinates (x, y) of the light location into the spherical coordinates (r, θ, ϕ) in the engine form for the light position in the Cartesian coordinates (x, y, z). The corresponding light position on the spherical environment map would be represented by a virtual light that has most of the features of the real light including angle, color, intensity, shadow strength and more.

The spherical coordinates are represented as follows, after normalization form Texel to sphere transformation.

$$(r, \theta, \phi) = \int_{1}^{lights} (\sqrt{x^2 + y^2}, \tan^{-1} \frac{y}{x}, \cos^{-1} \frac{z}{r}) \tag{1}$$

Where *lights* is the number of light detected and The measurement and dimensions of the panoramic 360° view affects the spherical projection as follows:

$$(dw, dh) = \int_{1}^{lights} (\frac{(360 \times 2 - 1)(x - 0)}{(w - 0) + 1}, \frac{(360 - 0)(x - 0)}{(w - 0) + 1}) \tag{2}$$

The virtual light location corresponding to the real light is represented as the resulting coordinate, if the light is placed onto the left side of the entire image frame.

$$(x, y, z) = \int_{1}^{lights} (\frac{\sin\phi \cos\theta}{\pi \times dh}, \frac{\cos\phi}{\pi \times dw}, \frac{\sin\phi \cos\theta}{\pi \times 90}) \tag{3}$$

However, if the real light is placed onto the right side of the entire view the location is represented as:

$$(x, y, z) = \int_{1}^{lights} (\frac{\sin\phi \cos\theta}{\pi \times (dh - 90)}, -\frac{\cos\phi}{\pi \times (dw - 180)}, \frac{\sin\phi \cos\theta}{\pi \times 90}) \tag{4}$$

Then, the light color is sampled based on the mean color of each contour. While the light intensity is influenced by the light area and the view dimensions, the shadow strength is applied based on the light intensity. The values are scaled to a suitable range for the 3D engine.

3.2 Simulate Reflected Light

In this section, two sub-methods are used to simulate the reflected light which embodies the surrounding environment for each virtual object in the scene.

Global Cube Map. This sub-method was discussed in our earlier work [24] but it is presenting briefly here for the context and further analysis. The cube map method also utilizes the panoramic 360° view to create 6 faces (left, back, right, front, top, bottom) which consist of 2D textures to construct a cubic HDR that can be assigned to the image-based lighting mode property. Information about the indirect illumination such as additional lights, real objects and surfaces are included in the 360° view which enrich the cube map surrounding the virtual object with details. The cube map is created from a live-feed every frame, therefore a coroutine concept is found to be useful to accelerate the rendering process. The unfolded cube is formatted from the appropriate six faces whose area is arranged typically in horizontal cross configuration. Thus, the cube faces would fit perfectly when the texture maps are folded.

Each face of the cube textures is saved as a separate image file then collected again in the arrangement. Each pixel in every face has a color that is calculated through the spherical projection then normalized for any further rotation and movements as follows:

$$(\theta, \phi) = \int_0^{points} (\tan^{-1}\frac{z}{x}, (\cos^{-1} z) + \frac{dir \times \pi}{180}) \quad (dx, dy) = \int_0^{points} (\frac{\theta}{\pi}, \frac{\phi}{\pi}) \quad (5)$$

The values should be maintained inside the height and the width of the created texture, to represent the final color as:

$$(px, py) = \int_0^{points} (dx \times w, py \times h) \qquad C = \begin{pmatrix} px \\ h - py - 1 \end{pmatrix} \qquad (6)$$

The resulted cube map is assigned to the image-based lighting mode when the shading properties are defined. Many other features can be manipulated in order to find the perfect material for each virtual object.

The cube map constructed was purely developed for 3D engine C# which allow reading from a live streaming video in real-time. There are many cube map tools available but having a previous video recording or a panoramic static picture are required for these tools.

Local Sampling. The creation of cube maps in every frame raises the performance cost, so we sample the region surrounding each virtual object from the view of the main camera which depicts a close-wide illumination information. A similar but short version was covered in our previous work [23] but got developed and improved to fit the local sampling approach.

Therefore, A plane attached to the virtual object is added to sample the live-video texture of the AR main camera. A mesh filter is initialized to keep the vertices of the mesh updated at 100 frames per second. A culling mask is used to hide the undesirable part of the view inside the object layer.

The mesh renderer of the background plane gets the main texture and assigns it to the video texture. The background plane is dispatched every 10 frames per second. Each vertex in the final mesh is transformed from the 3D world to view port point. The region capturing in progress the Vuforia device orientation is modified based on the screen orientation. Also, the local texture resolution is transformed based on the local scale of the entire camera output texture.

A virtual camera renderer captures the target texture to set the image-based lighting mode with that texture to define the material property of the virtual object.

The advantage of this sub-method is to sample only the local region on a specific object instead of the whole environment. It provides a suitable outcome for diffuse and glass materials, however the cube map provides better result with the specular materials.

3.3 Define Shading Properties

As known the shader is a collective computation of the shading properties during rendering. For a realistic object, the proper level of light, darkness, color must be considered. The virtual object consists of vertices, UV information and normals as part of the material features. Some of these properties are predefined such as the object's main texture and normal map, while other can be manipulated at the run-time like image-based lighting textures or cube maps.

In this system, two types of shaders are used to fully define the entire shading properties based on how the virtual objects receive the lights.

Lit Surface. The objects that receive and reflect the lights in the virtual scene have this type of shader which provides three passes: first forward base for the main light, second forward add for any additional lights, and third for casting the shadow on the other surfaces.

Also, it contains various properties such as vertex/fragment lighting mode, Ambient light, Lambert diffuse, Blinn Phong specular, Fresnel, image-based Lighting mode, Ashikhmin, Shirley and Premoze BRDF anisotropy.

Lambert Diffuse. It is calculated from the normal direction (N), light direction (L), color (c), diffuse factor (f) and attenuation (a), using the famous formula: $d = \sum_{i=0}^{n} c \times f \times a \times max(0, \overrightarrow{N} \cdot \overrightarrow{L})$.

Blinn Phong Specular. Some selective surfaces depict shining parts and lean towards representing some highlights. The calculation of the specular effect, required the mesh normal (N), light direction (L) with world space view direction (v), specular color (S_c), specular factor (S_f), attenuation (a) and specular power (S_p). At the beginning we must calculate the halfway vector direction (h) by normalizing the light direction (L) and world space view direction (v) represented in the model: $S = \sum_{i=0}^{n} S_c \times S_f \times a \times max(0, \overrightarrow{N} \cdot \overrightarrow{h})_p^S$.

Image-Based Lighting Reflection. The simulated reflected light in the form of a cube map or local 2d textures are assigned in this part where the required computation for each type is redefined based on the texture form. The texture will stay updated with live-feed video in real-time.

Image-Based Lighting Refraction. Same as the previous description except the received texture is bent to mimic the refraction effect. The Snell's law is used to represent the refraction model: $R = e \times i + [(e \cos_i \theta) - \sqrt{1 - e^2(1 - \cos_r^2 \theta)}]\overrightarrow{N}$ Where (i) is the velocity of light in vacuum and (r) is the velocity of light in the medium.

Image-Based Lighting Fresnel. It simulates the glass and water reflection/refraction where the viewer's point of view influences the normal vector. $Fresnel(F) = f + (1 - f)(1 - \overrightarrow{v} \cdot \overrightarrow{h})^5$.

Ashikhmin, Shirley and Premoze BRDF Anisotropy. A function that defines a certain pattern of how the light is going to hit a surface material such as silver or copper, The BRDF secular reflectively model used is:
$S = \frac{\sqrt{(n_u+1)(n_v+1)}(N \cdot h)^{\frac{n_u(H.T)^2 + n_v(H.B)^2}{1-(N.H)^2}}}{8\pi \times (v.H) \times max((N.L),(N.V))} \times F$
where (n_u, n_v) refer to width/height of tangent map, while (N) is normal vector, (h) halfway vector, (T) tangent vector, (B) BiTangent vector.

Unlit Surface. The object that only casts shadow without receiving any light are assigned to this shader. The background of the virtual scene should be hidden but must reflect the shadow of other virtual objects. Therefore, an alpha mask is used to cut-out the main color of the background plane to produce a transparent material lookalike that receives the shadow but nothing else.

4 Result Evaluation

The developed system is examined through multiple scenes with different lighting conditions and various environment settings. For further analysis, the results are evaluated based on three categories:

4.1 Incident Light Evaluation

Shadow observation is the most visible clue that can be used to determine the accuracy of the incident light estimation in our experiment, in addition to the light color and intensity of the whole scene. The shadow cast from the real objects in comparison with the shadow cast from the virtual objects is a tangible evidence that the estimated light angle is valid.

For the real objects, the angle of the real light is calculate manually by measuring the real object length (t) in the physical scene (see Fig. 3) with the aspect ratio to the shadow length (s) in six scenarios to obtain the measured angle $\theta = \tan^{-} 1\frac{t}{s}$. The measured angle θ then is compared to the estimated angle ϕ presented in the virtual light (y) axis where both angles provide a small margin of error.

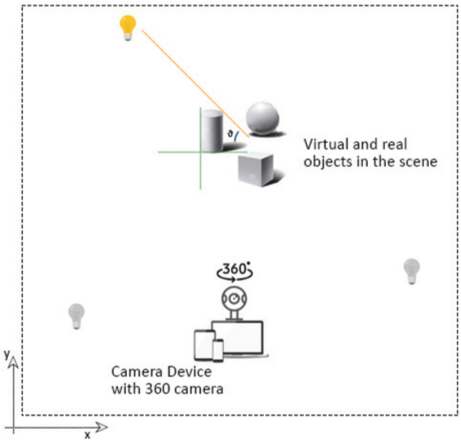

Fig. 3. Illustration of the final scene setting including the physical scene and the augmented elements.

Table 1 shows that the difference between the measured angle θ and the estimated angle ϕ is a reasonable amount which proves that our estimation is nearly accurate.

The computed statistics of the errors combined are averaged using the root-mean-square error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{6} Error_i^2}{6}} = 1.567 \tag{7}$$

Table 1. The error margin between the measured (θ) and estimated (ϕ) angles of the incident light in degrees.

Scene number	Shadow length (s)	Measured (θ)	Estimated (ϕ)	Error
1st	28	−167.04	−166.12	0.2
2nd	5.4	−50.19	−49.01	−1.18
3rd	15	27.47	26.24	1.23
4th	6	−127.56	−126.98	0.58
5th	18	203.42	203.01	0.41
6th	4	58.39	61.10	−2.71

4.2 Performance Evaluation

The reason for presenting two sub-methods for simulating the reflected light is to reduce the performance cost as mentioned above. In this section, we evaluate the performance through different scenes for both global cube map and local sampling.

Table 2. Performance evaluation for the global cube map against the local sampling with different number of objects

Scene no.	Global cube map			Local Sampling		
	1	2	3	1	2	3
FPS [1/s]	6	5	4	45	40	36
Update [ms]	173.4	177.3	183.2	35.9	37.8	40.2
Input [ms]	0.04	0.05	0.07	0.05	0.06	0.06
Surfaces [ms]	4.21	4.98	5.48	4.42	4.69	4.75
Camera render[ms]	0.66	0.68	0.84	1.08	1.40	1.69
Rendering [ms]	0.03	0.04	0.05	0.04	0.05	0.06

Table 2 shows that the number of virtual objects in each scene has limited influence on the performance cost. Although the local sampling reduces the performance cost significantly, the camera render is delayed by approximately 1 ms. In order to sample the required local area from the main view in certain regions related to the location of the virtual object, the performance cost has to be increased at a small fraction compare to the enhanced performance in the update function and other aspect of the system.

4.3 User-Feedback Evaluation

A user study is conducted as an online survey that contains several pictures and video of the results in different lighting conditions. The feedback data from 33 subjects who identified as college students shows that the incident light estimation is 51.2% more accurate than the average of faulty results. However, the angle of virtual objects shadow scored 20.4% higher than the estimation of the incident light. This observation is what led to the calculation of incident light angle in Sect. 4.1 above (see Fig. 4).

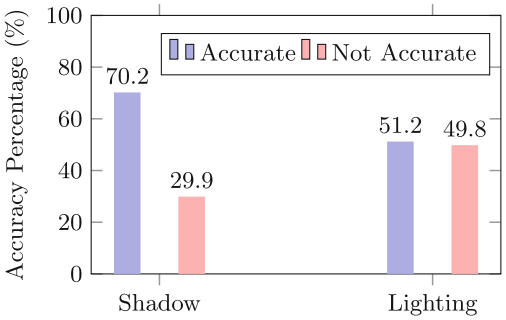

Fig. 4. User-feedback for the incident light estimation comparing to the shadow accuracy in general.

The results from the reflected light simulation method are not the same as shown (in Fig. 5) where the subjects are asked to evaluate which column presented more realistic results. The sampling of local regions had 66.67% more realistic results than the global cube map.

Furthermore, each object in the scene was rated based on how realistic they look compared to the other object whether it was real or virtual. The user ranked the object from 0 to 10 range as [(very virtual, 0), (virtual), (natural), (realistic), (very realistic, 10)].

Some users believed that some real objects were virtual by 1.5 points of the ranking which is a proof that the system works. While there is a slight misconception recognizing the virtual object from the real ones, the rating is also influenced by that misconception. Thus, the results are re-evaluated for the final feedback from the users based on the scene that provided the best realistic outcome according to the subjects' recommendations where all the virtual objects scores above the average (see Fig. 6).

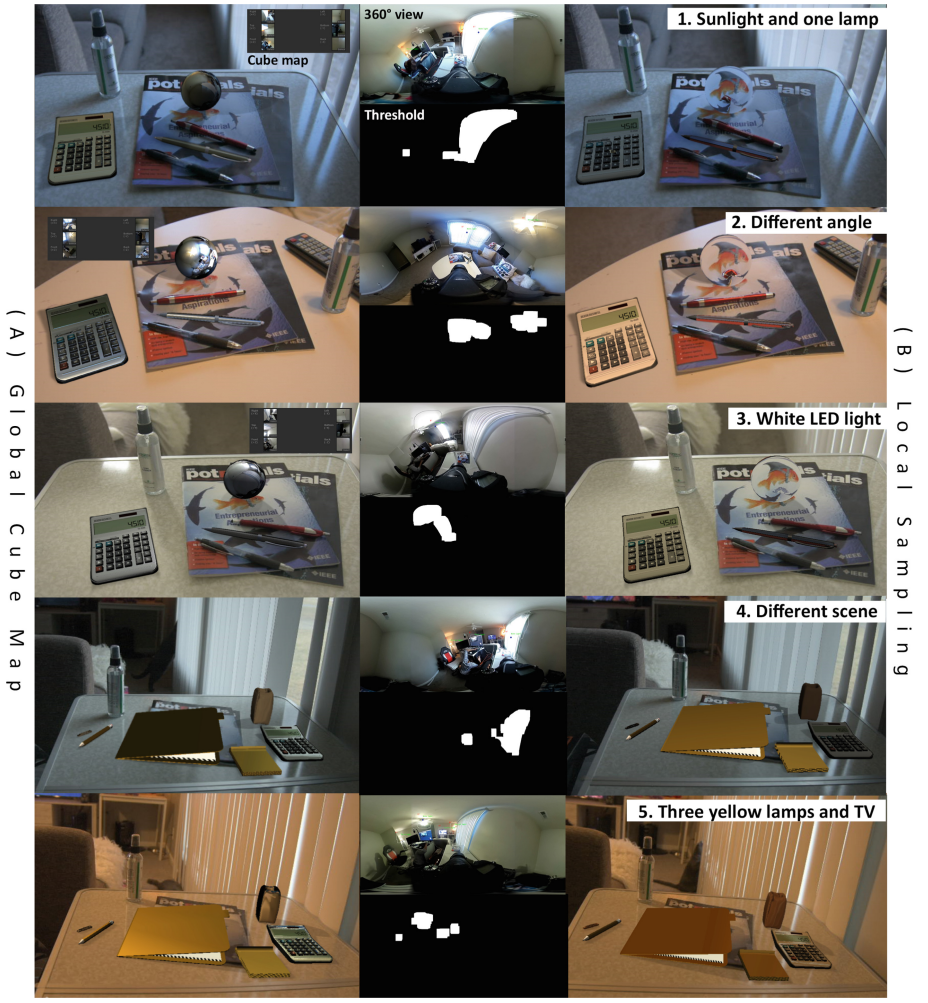

Fig. 5. System results where each environment condition has: (1) 360° view, (2) histogram-based thresholding, (3) cube map textures, (4) final scene of the global cube map creation result, and (5) final scene of the local sampling

The reflected light sub-methods influence the shading properties, and it seems that the global cube map creation provides more realistic results for the metal material objects where the local sampling has a more realistic outcome when the materials are glass or transparent. Finally, the lighting conditions are updated and changing in real-time accordingly, even when the object, camera or light are moving based their locations.

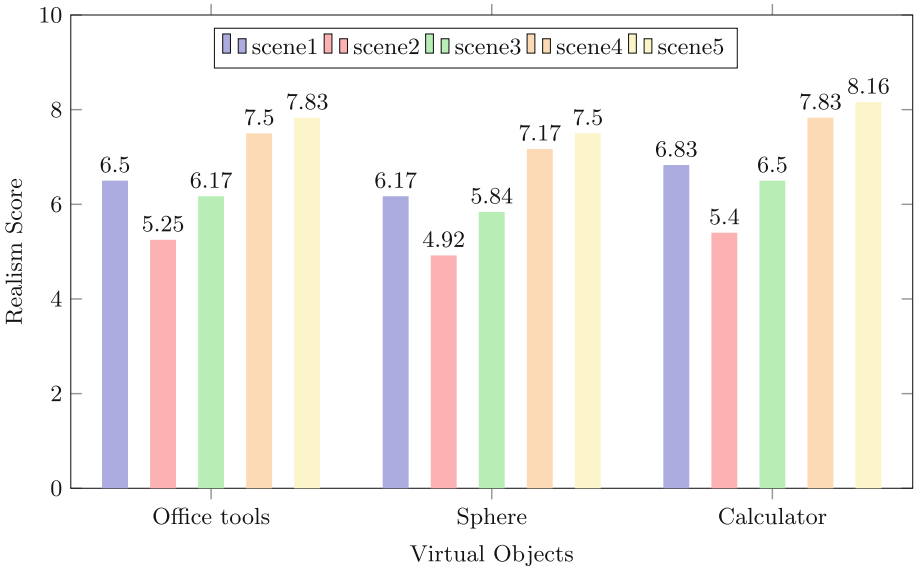

Fig. 6. Evaluating the virtual objects realism based on the scene environment and lighting condition.

5 Conclusion and Future Work

Light is an essential phenomenon to improve realism in augmented scenes, and we have been observing the light from different perspectives to estimate both incident and reflected light in the most practical methods. We are aware that our system can be improved in many ways, although current implementation generates plausible rendering. Estimating the incident light is the main goal to have the perfect light for any environment. The current method studies the light based on color brightness which is the most common method, however the physics-based lighting method is the focus for investigation in our future work.

Simulating the reflected light provides evidence from multiple evaluations that the global cube map is not an accurate method and reduces the overall performance, therefore, the local sampling can be explored further for a more realistic approach [25]. Also, a complete GPU implementation would provide a better performance cost in general. The experiment does not involve tracking development currently, but it can be improved in the future, as the tracking issue only appears in the video result where you can see the virtual object fluctuate sometimes.

It is reasonable to provide a step for hand gesture occlusion with the virtual elements, and real/virtual objects occlusion for more realistic view when the objects are moving in the final scene. Although the system solve the inter-reflection from the real object onto the virtual objects the reverse procedure where the virtual object is reflect onto the real object which required geometry

registration and mapping is under development for future work. Some physics effects such as caustics which required shadow and photon mapping is also under investigation.

References

1. Debevec, P.: Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1998, pp. 189–198. ACM, New York (1998)
2. Karsch, K., Hedau, V., Forsyth, D., Hoiem, D.: Rendering synthetic objects into legacy photographs. ACM Trans. Graph. (TOG) **30**(6), 157 (2011)
3. Agusanto, K., Li, L., Chuangui, Z., Sing, N.W.: Photorealistic rendering for augmented reality using environment illumination. Paper presented at the Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003 Proceedings (2003)
4. Fournier, A., Gunawan, A.S., Romanzin, C.: Common illumination between real and computer generated scenes. Paper presented at the Graphics Interface (1993)
5. Franke, T.A.: Delta light propagation volumes for mixed reality. Paper presented at the 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2013)
6. Gruber, L., Richter-Trummer, T., Schmalstieg, D.: Real-time photometric registration from arbitrary geometry. Paper presented at the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2012)
7. Jacobs, K., Loscos, C.: Classification of illumination methods for mixed reality. Paper presented at the Computer Graphics Forum (2006)
8. Kan, P.: High-quality real-time global illumination in augmented reality. Ph.D. thesis. Institute of Software Technology and Interactive Systems (2014)
9. Knecht, M., Traxler, C., Mattausch, O., Purgathofer, W., Wimmer, M.: Differential instant radiosity for mixed reality. Paper presented at the 9th IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2010)
10. Kronander, J., Banterle, F., Gardner, A., Miandji, E., Unger, J.: Photorealistic rendering of mixed reality scenes. Paper presented at the Computer Graphics Forum (2015)
11. Mehta, S. U., Kim, K., Pajak, D., Pulli, K., Kautz, J., Ramamoorthi, R.: Filtering environment illumination for interactive physically-based rendering in mixed reality. Paper presented at the Eurographics Symposium on Rendering (2015)
12. Nowrouzezahrai, D., Geiger, S., Mitchell, K., Sumner, R., Jarosz, W., Gross, M.: Light factorization for mixed-frequency shadows in augmented reality. Paper presented at the 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2011)
13. Pessoa, S., Moura, G., Lima, J., Teichrieb, V., Kelner, J.: Photorealistic rendering for augmented reality: a global illumination and BRDF solution. Paper presented at the 2010 IEEE Virtual Reality Conference (VR) (2010)
14. Richter-Trummer, T., Kalkofen, D., Park, J., Schmalstieg, D.: Instant mixed reality lighting from casual scanning. Paper presented at the 2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2016)
15. Rohmer, K., Büschel, W., Dachselt, R., Grosch, T.: Interactive near-field illumination for photorealistic augmented reality with varying materials on mobile devices. IEEE Trans. Vis. Comput. Graph. **21**(12), 1349–1362 (2015)

16. Schwandt, T., Broll, W.: A single camera image based approach for glossy reflections in mixed reality applications. Paper presented at the 2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (2016)
17. Sloan, P.-P., Kautz, J., Snyder, J.: Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. Paper presented at the ACM Transactions on Graphics (TOG) (2002)
18. Supan, P., Stuppacher, I., Haller, M.: Image based shadowing in real-time augmented reality. IJVR **5**(3), 1–7 (2006)
19. Unger, J., Gustavson, S., Ynnerman, A.: Spatially varying image based lighting by light probe sequences. Vis. Comput. **23**(7), 453–465 (2007)
20. Rhee, T., Petikam, L., Allen, B., Chalmers, A.: MR360: mixed reality rendering for 360 panoramic videos. IEEE Trans. Vis. Comput. Graph. **23**(4), 1379–1388 (2017)
21. Iorns, T., Rhee, T.: Real-time image based lighting for 360-degree panoramic video. In: Huang, F., Sugimoto, A. (eds.) PSIVT 2015. LNCS, vol. 9555, pp. 139–151. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30285-0_12
22. Michiels, N., Jorissen, L., Put, J., Bekaert, P.: Interactive augmented omnidirectional video with realistic lighting. In: De Paolis, L.T., Mongelli, A. (eds.) AVR 2014. LNCS, vol. 8853, pp. 247–263. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13969-2_19
23. Alhakamy, A., Tuceryan, M.: AR360: dynamic illumination for augmented reality with real-time interaction. In: 2019 IEEE 2nd International Conference on Information and Computer Technologies ICICT, pp. 170–175 (2019)
24. Alhakamy, A., Tuceryan, M.: CubeMap360: interactive global illumination for augmented reality in dynamic environment. In: IEEE SoutheastCon (2019, accepted and presented)
25. Alhakamy, A., Tuceryan, M.: Polarization-based illumination detection for coherent augmented reality scene rendering in dynamic environments. In: Proceedings of ACM Computer Graphics International (2019, accepted)